International Journal of Science and Research (IJSR)
ISSN (Online): 2319-7064
Index Copernicus Value (2013): 6.14 | Impact Factor (2013): 4.438

Survey Paper on Location Privacy Preserving and Finding Optimal Meeting Location using Secured Homomorphism

Ritesh Thakur[1], Vilas Shelke[2]

[2]Savitribai Phule Pune University, Institute of Knowledge COE, Pimple Jagtap, Pune, Maharashtra, India

[1]Savitribai Phule Pune University, Institute of Knowledge COE, Pimple Jagtap, Pune, Maharashtra, India

Abstract: ***Equipped with progressive smartphones and mobile devices, today's extremely interconnected urban population is more and more addicted to these gadgets to prepare and set up their daily lives. These applications usually suppose current (or preferred) locations of individual users or a bunch of users to produce the required service, that jeopardizes their privacy; users don't essentially wish to reveal their current (or preferred) locations to the service supplier or to different, presumably untrusted, users. During this paper, we tend to propose privacy-preserving algorithms for deciding Associate in Nursing best meeting location for a bunch of users. we tend to perform a radical privacy analysis by formally quantifying privacy-loss of the projected approaches. so as to check the performance of our algorithms during a real readying, we tend to implement and take a look at their execution potency on Nokia smartphones. By means that of a targeted user-study, we tend to plan to get Associate in nursing insight into the privacy-awareness of users in location based mostly services and also the usability of the projected solutions.***

Keywords: *privacy*, Mobile application, *oblivious computation.*

1. Introduction

The rapid abundance of smartphone technology in urban communities has enabled mobile users to utilize context aware services on their devices. Service providers take advantage of this dynamic and ever-growing technology landscape by proposing innovative context-dependent services for mobile subscribers. Location based Services (LBS), for example, are used by millions of mobile subscribers every day to obtain location-specific information [1].

Location privacy preservation in mobile environment is challenging for two reasons. Firstly wireless communications are easy to intercept e.g. eavesdropper can collect transmitted information of mobile users at certain public place. Besides, since people are publicly observable, context information can easily be obtained from their conversation or behaviors. As a result, partial trajectory information associated with a user's real identity is inevitably exposed to the eavesdropper.

Second, the limited resources of mobile devices greatly restrict Privacy Enhancing Technologies one could apply and deploy in wireless network. Current solutions reply on simple schemes to hide the real identity of a mobile user from a passive adversary, rather than complex cryptographic technologies.

Two popular features of location-based services are location check-ins and location sharing. By checking into a location, users can share their current location with family and friends or obtain location-specific services from third-party providers. The obtained service does not depend on the locations of other users. The other types of location-based services, which rely on sharing of locations by a group of users in order to obtain some service for the whole group, are also becoming popular.

Privacy of a user's location or location preferences, with respect to other users and the third-party service provider, is a critical concern in such location-sharing-based applications. For instance, such information can be used to de-anonymize users and their availabilities, to track their preferences or to identify their social networks. In the taxi-sharing application, a third-party service provider could easily deduce home/work location pairs of users who regularly use their service. Without effective protection, if the collected data is leaked in an unauthorized fashion or improperly shared with corporate partners, which could have severe consequences on the users' social, financial and private life.

2. Privacy Preserving Route Planning

This paper proposed Secure Multiparty Computation (SMC) protocols for securely computing the distance between a point and a line segment, the distance between two moving points and the distance between two line segments.

2.1 Route Planning Protocols

three types of protocols for route planning including: i) the distance between a route (line segments) and a set of fixed objects (points), ii) the distance between two moving objects with constant velocity (parameterized lines), and iii) the distance between a route (line segments) and another route (line segments).

2.1.1 Points and Line Segments

Suppose Alice has a set of points each associated with a distance and Bob has a route which consists of a set of lines.

Furthermore, Alice and Bob want to know if Bob's route gets too close to a point in Alice's set. Note that the protocols can easily be extended to have the threshold associated with the segments instead of the points.

2.1.2 Parameterized Lines

Suppose Alice has a route which consists of several line segments and Bob has a route which consists of a set of line segments. Furthermore, suppose that there is an object that is traveling on each of Alice and Bob's routes whose position can be described by a point with constant velocity (with parametric equations) and that Alice has a threshold distance for which she does not want her object to get close to Bob's object.

3. Shall we meet? Proposing optimal locations for meetings

Presented a survey of existing literature on meeting-location algorithms and propose a more comprehensive solution for such a problem. The list of participants, the proposed meeting time, likely start locations and possible travel methods are known. The "cost" function (time, distance, social constraints, etc.) for each person to travel to locations are calculated. Although considering aspects such as user preferences and constraints, their work does not address any security or privacy issues.

- Distances between meridians vary. Given latitude, significant errors can occur when participants are located away from the equator.
- Global curvature is important. While this may be acceptable when participants are within a few kilometers of each other, it's a poor solution for participants spread over larger distances.
- Participants incur different costs while traveling. For
- Example, if a high-wage person is meeting with a low-wage person, it would be cheaper to request the lower-wage person go to a location closer to the higher earner than to a "fair" spot.
- Hemispheres matter (for example, if one participant is in Tokyo and another in Seattle). The system's result may be impractical. Simple averaging would suggest a meeting in Corsica, which would be inconvenient for both, instead of a location in Alaska or Hawaii that might be better.
- Local geography, such as roads, paths, and modes of transport are important. For example, in Rio de Janeiro, if one participant was at the international airport and another in Niterói, the system might propose a meeting on the southeast end of the Ilha do Governador, not realizing that the participant coming from Niterói would drive past the airport on the way to the Ilha do Governador? (See Fig. 3.1)

3.1 Scenario

Friends together, imagine that five friends want to get together at a restaurant. Two will be leaving from their workplaces, another is arriving at the local airport, one is finishing class at the university, and another will be leaving from home. Some will be driving, some will be taking public transportation, and some will have a choice. And all can walk. They want to get together immediately to eat. There are over 1,000 restaurants in this metro area, and many are acceptable to all. Where shall they go Business Meeting**?** Imagine that a two-day business meeting is planned at a large company. Dates are set. Participants will arrive from all over the world. It is essential that the meeting occur at one of the company's facilities. The company wants to minimize the total expense (this includes travel costs and employee time.) Participants will either fly or drive to the site. If some participants already work at a location that can host the meeting, they won't have to travel at all. At which company location should the meeting take place?

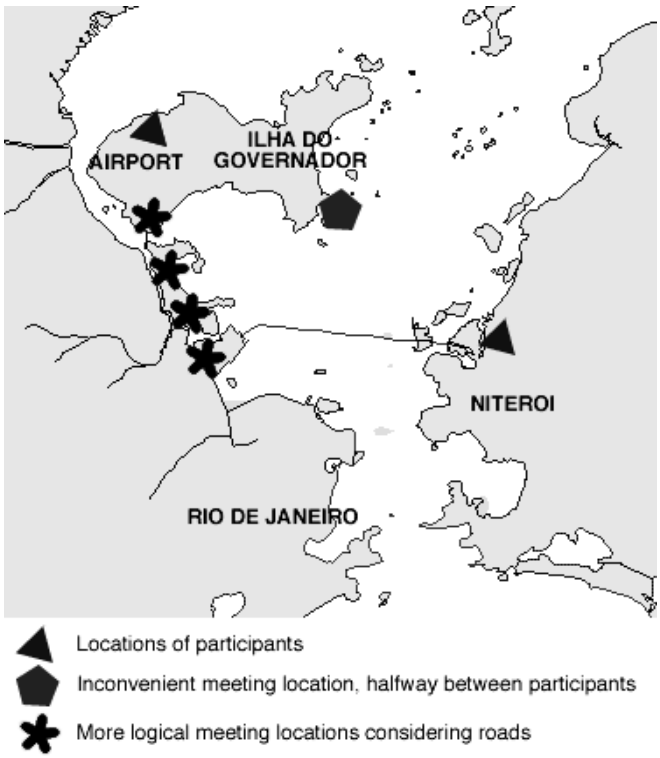


Figure 3.1: Optimization for geographic centers might miss logical meeting points

3.2 Finding the Optimal Location

For both scenarios, there are inputs and outputs for our meeting location finder. Inputs include:

- A participant set (friends; employees) and their start locations (where friends are starting; where employees work or live)
- A set of candidate locations (the cafes in a metro area; a list of company facilities)
- The modes of transportation available to each (walk, rive, bus, fly, drive, etc.)
- The desired function to be minimized (the time that the last friend will get to the restaurant, i.e., the maximum time for any participant to get to the restaurant; the total cost to the corporation)
- Other criterial functions (time; dollars, profiles with social constraints)

The output is proposed locations for the meeting. The output may be one location only, or a list of proposed locations rank-ordered by a function that is minimized. Such a list may be short or long, depending on variables.

4. Trust no one: A decentralized matching service for privacy in location based services

Suggested a novel approach to deploy location-based services in which user privacy is guaranteed. without any entity having knowledge of both pieces of sensitive user information i) location ii) queries (interests + social relationships). Inspite of operating on encoded information got from the operator and the LBS; it is able to trigger updates to the mobile user whenever the user is in the same location of its interested services.

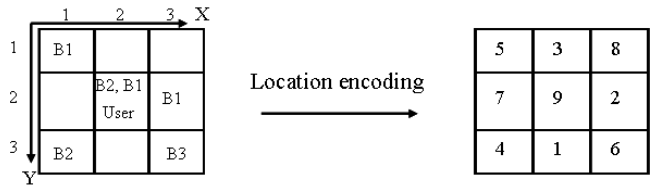


Figure 4.1: Pseudonymized locations (PLs) of a 2-D location space

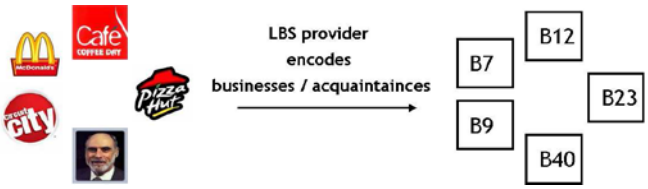


Figure 4.2: Pseudonymized identifiers (PIs) of b usi-nesses/acquantainces

4.1design: A Decentralized Matching Service

We now describe an approach wherein the matching of an end-user's location and their interests in a LBS is carried out by an entity unaware of the actual value (or identification) of both the location and the interests. This is done by encoding the two pieces (location, business-interests / social-acquaintances) of sensitive information in symbol space, and Designing a matching service that solely operates on the encoded values, and informs the user of nearby services of interest. The encoding of the two pieces of sensitive information is done as follows:

- Pseudonymized Locations (PLs) the location information of the mobile user is either coarse-grained information (via cellular triangulation or cell-ID localization) known to the mobile operator or fine-grained information (via GPS-enabled phones if available) known to the mobile user. Encoding the location information, as shown in Figure 3.1 is done by partitioning the physical location space into say 2-D location grids of size grid-size (say 100 m), and assigning a pseudonymized location (PL), a number in the range [0,N] to each grid. We require some entity to assign PLs. Given that the mobile operator already has coarse-grained information of every mobile user, the mobile operator is in the best position to serve as this entity assigning PLs mobile users equipped with GPS-enabled phones can transfer their coordinates to the mobile operator and get assigned PLs corresponding to their current location.

- Pseudonymized Identifiers (PIs) The LBS provider has a list of all businesses and users which are part of the LBS service. To reach the LBS provider assigns an

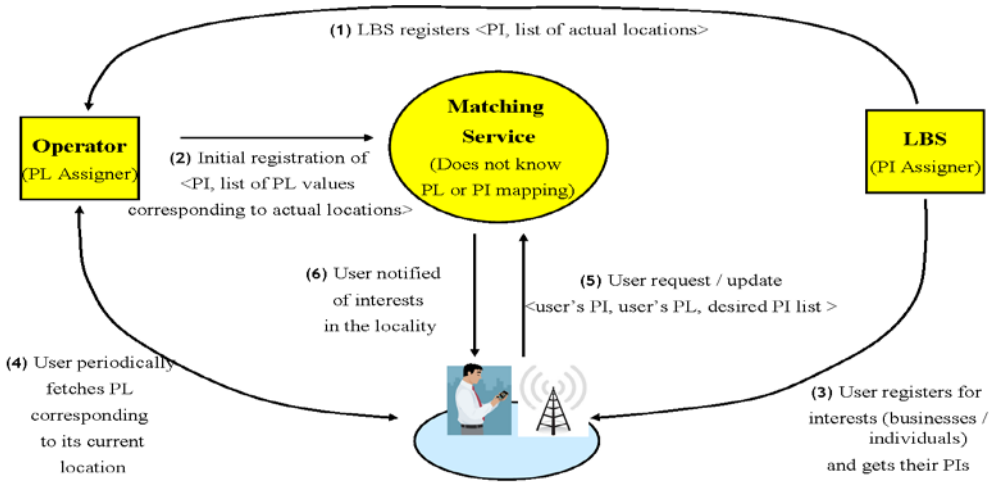


Figure 4.3: Architecture of matching service.

We now describe the design of the matching service that takes encoded inputs from the mobile operator and the LBS Service, and triggers the user when entities of interest are near its current location. The operator creates a mapping from actual locations to PLs for the addresses of businesses (and the current locations of users). Similarly, the LBS provider creates PIs from the identities of businesses and users (PI assignment for users are used for locating social acquaintances). Both these mappings are periodically refreshed every T minutes (say typically 15 mins) in order to avoid collusion attacks. As shown in Figure 4.3, in the boot-strap phase, the set of PIs corresponding to businesses, and

their encoded location PLs are fed into the matching service. This is done in the following steps - first the LBS notifies the mobile operator of the physical locations for each encoded business PI it caters to (indicated as Step 1 in Figure 4.3), followed by the mobile operator notifying the matching service of each PI with the corresponding PL values for each of the locations of that business (Step 2). The mobile user also relies on an initial registration phase with the LBS to get the PIs corresponding to the businesses and users of interest (Step 3). Before sending a LBS query, the mobile user contacts the mobile operator for the PL corresponding to its current location (Step 4). It can then initiate a LBS query, the generic form of which is a 3-tuple <user's-PI, user's-PL, desired-PI-list> (Step 5). Note that the 'user's PI' field is required only when the mobile user wants its social-acquaintances to locate them, else this field could be NULL.

The role of the matching service is then only to check if there are any PIs of interest to a user that are located in a location grid with the same PL as the mobile user, and if yes, create a trigger to the end-user app notifying of nearby PIs of interest (Step 6). Such a matching can be trivially done, since the matching service knows the set of PLs in which a particular PI is located. Note that because of initial registration phase of the mobile user with the LBS in Step 3, the mobile user can decide which businesses / acquaintances correspond to the notified PIs. Note the matching service knows of (and interacts with) only the PI and PL values of the various entities - never their actual names or locations.

4.2The need for a Matching Service

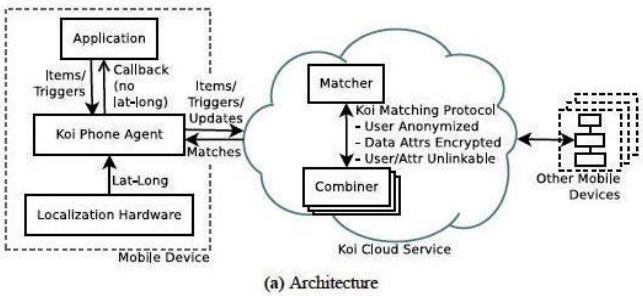
The goal of our paper was to design a system in which no one entity has access to all pieces of sensitive information related to the usage of location based services. Is it possible for entities in a LBS system (mobile user, LBS provider, mobile operator) to encode and exchange information they possess, in a manner that removes the need for a separate matching service? We consider such alternatives, and build a case for why a matching service is essential. LBS exchanges PIs directly with the mobile operator: In this approach, the LBS encodes businesses with PI encodings and notifies the mobile operator of the actual locations corresponding to each PI. The mobile user (as in Step 3 of Figure 4.3) registers with the LBS for interested services and gets the corresponding PIs of its interests. The user's query with the <desired-PI-list> is sent directly to the mobile operator. The mobile operator, based on the current location of the user, returns the PIs that are located in the vicinity of the user. In this case, the mobile operator, based on the PI/location mappings, and the user locations (and their PIs of interest) carries out the match between an user and their interests. However, the mobile operator can decode the PI mapping and infer the user's objects of interest. Suppose the operator has access to a yellow page service though which it can find businesses at any particular location. Consider a business with PI = 385, present at locations L = {L1,L2,L3, ...Ln}. The operators can find the set of businesses B(Li) corresponding to any location Li. Then, if Ti=n i=1 B(Li) corresponds to a single entity (i.e. cardinality of set is one), it can decode the business/interest corresponding to PI = 385. Mobile operator exchanges PLs directly with the LBS: In this approach, there are no business / acquaintance encodings (i.e. no PIs). The operator however assigns location encodings (i.e. PLs) to each location-grid. In order to match businesses, the LBS provider initially gives the mobile operator a list of businesses and their locations, and the operator returns to the LBS, a set (permuted to ensure it does not trivially reveal the PI to location mapping) of PLs associated with each business. In order to locate nearby businesses, a user sends it's PL (that it acquired from the operator) to the LBS, and is notified of businesses whose PL-set contains the user's current PL. In this approach however, the LBS provider can infer the location of an user, by correlating the PL sets corresponding to different businesses, and since it has knowledge of the physical locations of businesses. As shown in Figure 4.1, suppose B1 and B2 share a common location (here (x=2, y=2)). Since the only common entry in PL sets {9, 2, 5} corresponding to B1 and {4, 9} for B2 is 9, the LBS can decipher that the location having PL of 9 is (x=2, y=2).

5. A Location-Privacy Platform for Smartphone Apps

Proposed a privacy-preserving location based matching as a fundamental platform primitive and as an alternative to exposing low-level, latitude-longitude coordinates to applications. Applications set rich location-based triggers and have these be fired based on location updates either from the local device or from a remote device. But issue pertains to malicious applications registering a large number of triggers at sensitive locations, and reverse-engineering a victim user's location based on triggers matched. A weak defense against this attack would be rate-limit to the number of trigger registrations from an application.

5.1 System Architecture

Koi comprises two components, one which runs on the User's mobile device and the other in the cloud (Figure a). The mobile component of Koi interfaces between applications and the cloud component. To applications, it Exposes a simple API (Figure b), which allows registration and updating of *items* and *triggers*, and provides Notification through a callbacks. An *item* is a statement.



(a) Architecture

Registering Items and Triggers
ITEM = CREATEITEM(CONTENT, TTL)
TRIG = CREATETRIGGER(CALLBACK, TTL)
Create a new item or trigger.
Adding Attributes
ADDLOCATTR(ITEMORTRIG, LOC, AUTOUPDATE)
ADDAPPATTR(ITEMORTRIG, ATTR)
Associate location or other attributes to items and triggers.
Notifications
CALLBACK.NOTIFY(CONTENT)
Called by platform when a match is found.

(b) API

5.2 Platform API

The platform service model is similar to a database trigger. The app registers items with the Koi phone-agent. Items may correspond to users or content (e.g., photos). The app associates attributes with an item. Attributes may be locations, keywords, or arbitrary data. The app also registers triggers. Triggers specify one or more attributes that must match. When an item matching the trigger is registered (by another user or app, or by the same app), the app registering the trigger is notified of the item through a callback. Figure b lists the Koi API. The app specifies location attributes symbolically (e.g., loc: self, or within 1 block of loc: self). The Koi phone-agent internally replaces loc: self with the actual at-long. The agent optionally automatically updates the lat-long if the user's location changes. This amortizes the Cost of acquiring user location across multiple apps, and avoids having to wake each app up whenever the user moves. By never exposing lat-long data to the app, Koi minimizes the app accidentally leaking it to third-parties. The app may associate arbitrary content with an item. A social networking app may, for instance, register the user's push-notification service handle so another user can contact this user. A citizen-journalism app may, for instance, register a photo or URL etc. The content may additionally be encrypted, for instance in the social networking app, only friends with the appropriate key may recover the push-notification handle.

4.2 Privacy-Preserving Matching Service

The operation of the Koi cloud service is best illustrated through an example. Consider the scenario in Table 1 where users Alice and Chuck register an item indicating they are tour-guides for Bangalore and Boston respectively. Bob registers a trigger looking for a tour-guide at his present location. The goal is to match Bob, who happens to be in Bangalore, with Alice. Note that Bob's phone-agent uses his lat-long without first geocoding it to Bangalore. For simplicity, we assume each of them register some unique user ID (as the item content, or the trigger callback) through which they can be reached. Our location-privacy goal is to prevent the cloud service from associating this user ID with the user's location.

6. Analysis of all the Review Techniques

Privacy preserving route planning [2] proposed Secure Multiparty Computation (SMC) protocols for securely computing the distance between the points. One difficulty with route planning protocols is the requirement that the device know where it is at, which would seem to require some form of query to a GPS system, but this would reveal the location of the device.

Where shall we meet? Proposing optimal locations for meetings [3] proposing optimal locations for meetings presented a survey of existing literature on meeting-location algorithms and proposes a more comprehensive solution for such a problem. The system, while useful, may be complicated for some users. Automating system defaults when users provide insufficient data from calendars or start points can help, but preferences about times, venues, and travel methods can be complicated even when known. An organizer, or participants who vote, need to evaluate choices and fine-tune results to suit group criteria.

Trust no one [4] a decentralized matching service for privacy in location based services," suggested a novel approach to deploy location-based services in which user privacy is guaranteed. Inspite of operating on encoded information got from the operator and the LBS; it is able to trigger updates to the mobile user whenever the user is in the same location of its interested services.

Koi: A location-privacy platform for smartphone apps,[5] proposed a privacy-preserving location based matching. But issue pertains to malicious applications registering a large number of triggers at sensitive locations, and reverse-engineering a victim user's location based on triggers matched. A weak defense against this attack would be rate-limit to the number of trigger registrations from an application.

We plan to improve the execution time of PPFRVP and to investigate more efficient designs that reduce communication between devices and interactions between users, without sacrificing security or usability. We also plan to investigate ways to recover from errors and attacks such that after detecting an error in a subgroup, the entire group does not need to rerun the protocol.

References

[1] Igor Bilogrevic, Murtuza Jadliwala, Vishal Joneja, Kübra Kalkan, Jean-Pierre Hubaux, and Imad Aad, "Privacy-Preserving Optimal Meeting Location Determination on Mobile Devices" IEEE Transactions on Information Forensics and Security, vol. 9, no. 7, pp. 1141-1156, JULY 2014.
[2] K. B. Frikken and M. J. Atallah "Privacy preserving route planning," in Proc. ACM WPES, pp. 8– 15, 2004.
[3] P.Santos and H.Vaughn, "Where shall we meet? Proposing optimal locations for meetings," in Proc. MapISNet, 2007.
[4] S. Jaiswal and A. Nandi, "Trust no one: A decentralized matching service for privacy in location based services," Proc. ACM MobiHeld, 2010.
[5] S. Guha, M. Jain, and V. Padmanabhan, "Koi: A location privacy platform for smartphone apps," Proc. 9th USENIX Conf. NSDI, 2012.